13 juin 2010

H_DigiPLC

Manuel d'utilisation

Historique :

Date	Version	Auteur	Description	
01 juin 2009	1.0	Henri Laidet	Création	
25 juin 2009	1.1	Henri Laidet	Ladder :ajout action sur les entiers (move, add…)	
4 sept 2009	1.2	Henri Laidet	Ajout communication série avec ARM7	
1 fevrier 2010	1.3	Henri Laidet	Ajout compatibilité avec DigibutlerK3 sous MQX	
12 juin 2010	1.4	Henri Laidet	Adaptation dialogue série via Bluetooth (sceptre)	

Généralités:

Le programme est écrit avec « ***Microsoft Visual C# 2008 Express Edition*** ».

H_DigiPLC sert au dialogue entre PC et DigiButler (DB) via une com TCP/IP.
Depuis la version 1.2, le dialogue se fait aussi entre PC et ARMée via une com liaison série.
Depuis la version 1.4, le dialogue se fait via Bluetooth profil liaison série (carte ARM7 sceptre).
Pour échanger des fichiers, utiliser le Datalogger, créer et exécuter des programmes automate.

Digibutler :
- ColdFire MCF52231
- 1x Ethernet RJ45
- 1x Port série RS232 DE9

Armée :
- ARM7TDMI LPC2106
- 1x Port série RS232 DE9
- 1x Port série RS232 ou RS485 DE9

Sceptre :
- ARM7TDMI LPC2148
- 1x Port série FTDI USB
- 1x Port série Bluetooth

Fenêtre principale:

Environnement MDI

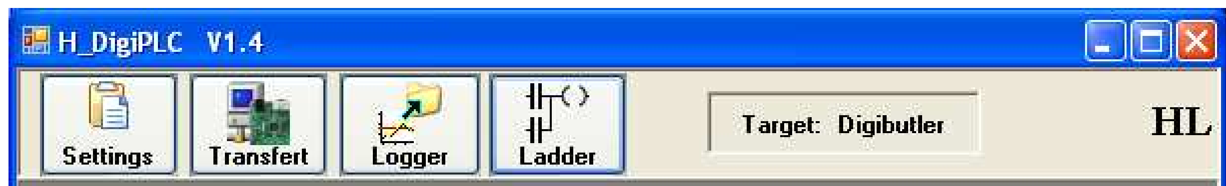


Figure 1

Settings:

Configuration : Assignation et sauvegarde
C# ne permet pas l'accès à la base de registre, tout est dans le fichier « ***HDB_settings.xml*** ».

Les champs T1 et T2 (Figure 2) permettent de changer de cible

- Champ T1 : Cible Digibutler sert aux spécificités coldfire (ports 8 bits, BigEndian)
- Champ T2 : Cible ARM7 idem T1 mais pour ARM7 (port 32 bits, LittleEndian)

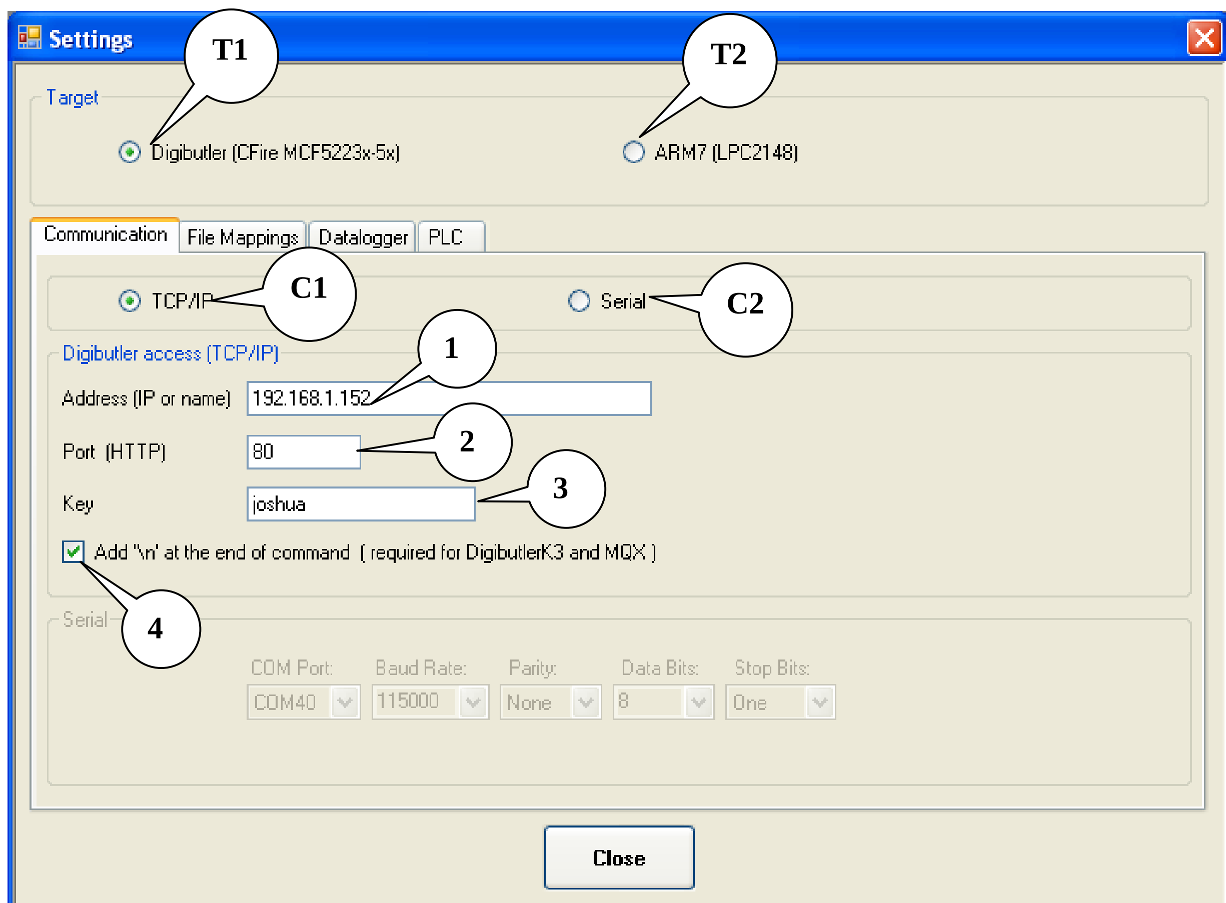


Figure 2

Communication :

Les champs C1 et C2 (Figure 2) fixent le type communication

- Champ C1 : Communication en TCP/IP (Digibutler)
- Champ C2 : Communication via port série (ARM7)

Digibutler access (TCP/IP):

Configuration du dialogue TCP/IP avec DB

- Champ 1 : Adresse IP Digibutler (***192.168.1.151*** ou ***monadresse.dyndns.org***).
- Champ 2 : Port du serveur http
- Champ 3 : Clé d'accès au Tunnel FTP. Permet la navigation dans les fichiers DB
  [**`ipport.h:    #define UPLOAD_KEY "joshua"`**]
- Champ 4 : Dialogue avec DigibutlerK3 (serveur HTTP MQX)

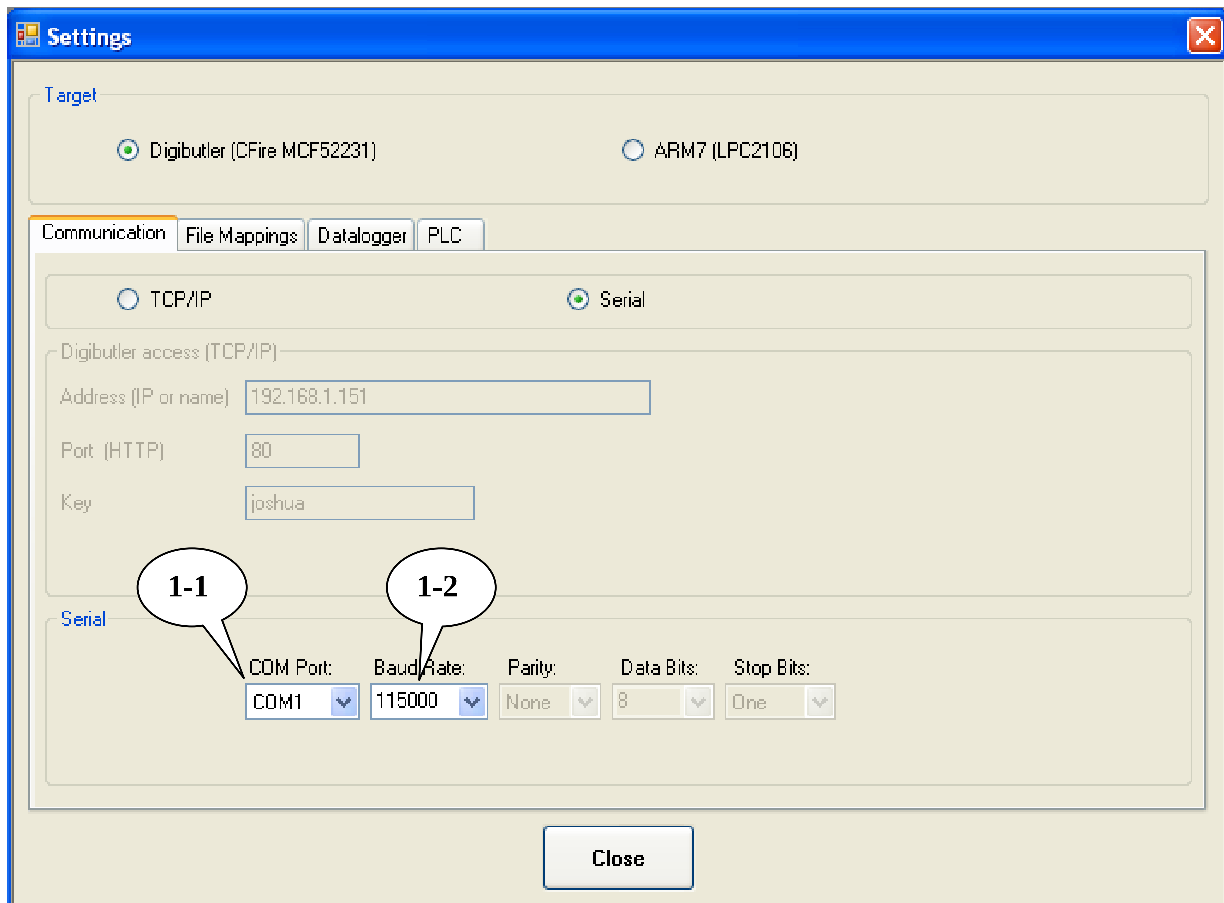


Figure 2-1

Serial :

Configuration de la communication sur port série (Figure 2-1)

- Champ 1-1 : Sélection du port de communication
- Champ 1-2 : Fixe le baudrate

Parité, data bits et stop bits sont imposés et non modifiables.

File Mappings :

Association des extensions de fichiers avec un exécutable (Figure 2-2).
Permet d'ouvrir les fichiers DB ou PC. (Dans la fonction ***Transfert*** bouton ***Show***).

- Champ 5 : Liste des extensions associées à un type de fichier (champ 6).
- Champ 6 : Types de fichiers (5 en tout IMAGE, TEXT, EXCEL, PDF, libre).
- Champ 7 : Extension associée
- Champ 8 : Exécutable lié à l'extension champ 7
- Champ 9 : Ajoute un ensemble Type, Extension, Exécutable à la liste champ 5
- Champ 10 : Supprime un élément de la liste champ 5

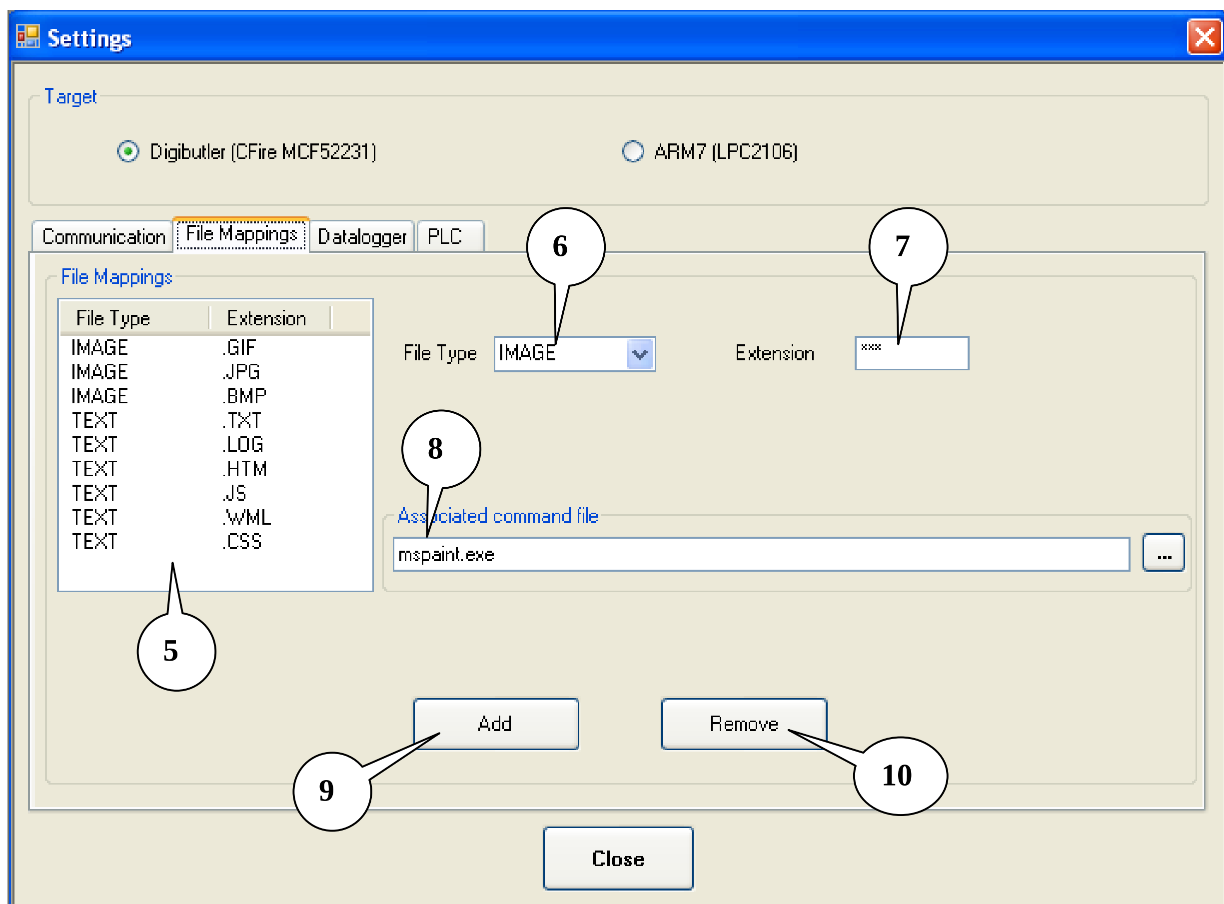


Figure 2-2

Datalogger :

Configuration de l'accès au Datalogger (Figure 2-3)

- Champ 11 : Nom du fichier de configuration sur DB
  [**Datalogger.c: #define DATA_LOGGER_CONF_NAME "config.cfg"**]
- Champ 12: Offset de correction heure par rapport au GMT

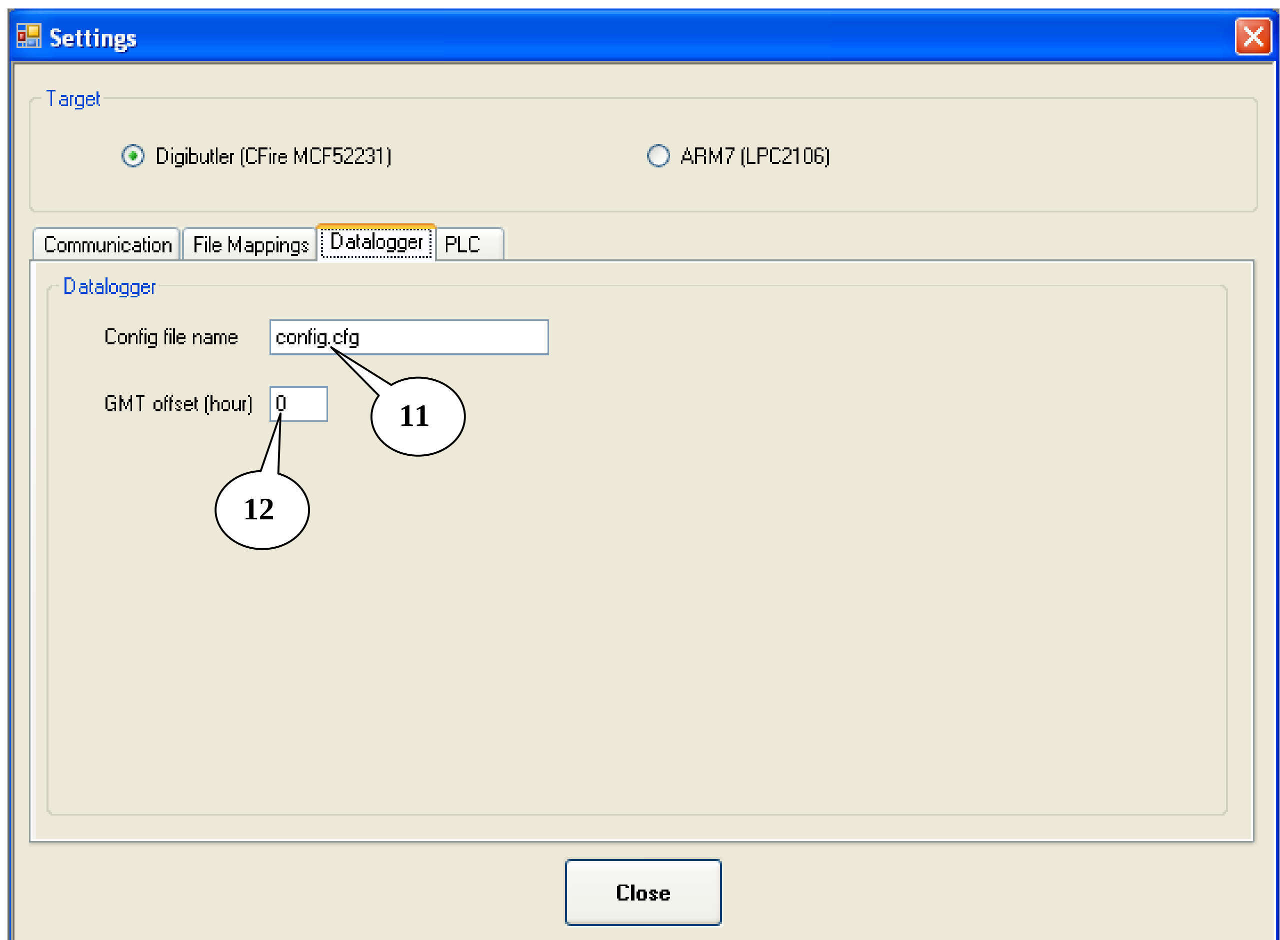


Figure 2-3

PLC :

Configuration du PLC (Figure 2-4)

- Champ 13 : Nom du fichier sur DB. C'est le fichier compilé utilisable par le moteur. [`PLC.h: #define PLC_FILENAME "plc.txt"`]
- Champ 14: Nom du fichier graphique sur le PC. C'est le fichier source du Ladder.
- Champ 15 : Port principal : masque des bits utilisés comme IO TOR
  8 bits (GPT0-3) sur Digibutler ou 32 bits (port0) sur ARM7
- Champ 16 : Port secondaire : masque des bits utilisés comme IO TOR
  8 bits (entrées Ana) sur Digibutler ou inexistant sur ARM7 (LPC2106)
- Champ 17 : Nombre d'expanders (utilisé pour proposer la liste des IO TOR).
- Champ 18 : Nombre de Bits d'usage général.
- Champ 19 : Nombre de Temporisations.
- Champ 20 : Nombre d'entiers (32 bits signés) d'usage général.

Lors de la connexion PC ⇔PLC, si les champs 15 à 20 ne correspondent pas, il y a demande de mise à jour automatique.

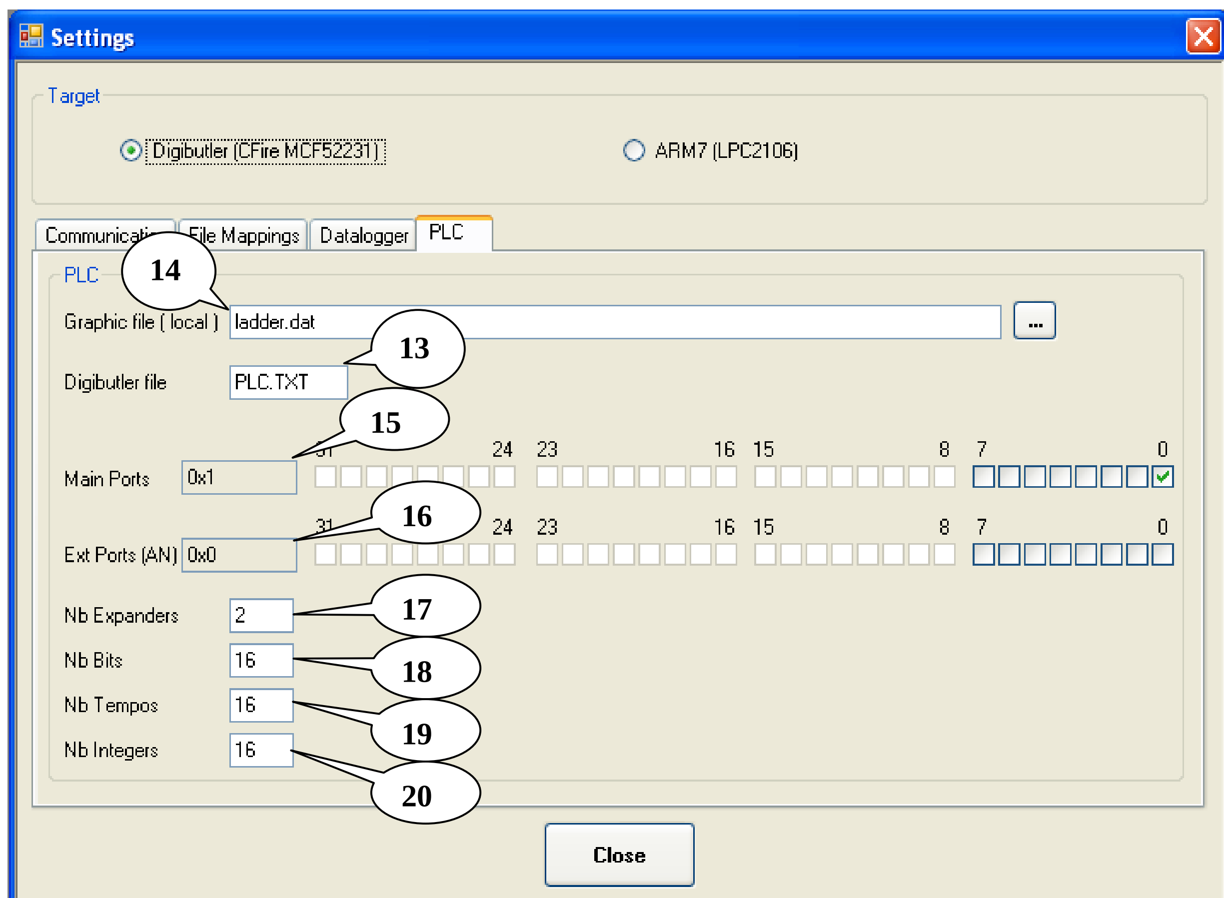


Figure 2-4

Modification de paramètres :

Si on modifie un paramètre, lorsqu'on quitte le programme on a un message



Figure 3

La réponse ***Oui*** sauvegarde le fichier XML avant de quitter.

Transfert :

Fonctionnalité implémentée sur DigiButler et ARM7.

Permet de lire et d'écrire les fichiers de Digibutler (ou ARM7) stockés sur carte SD.

Déconnecté :

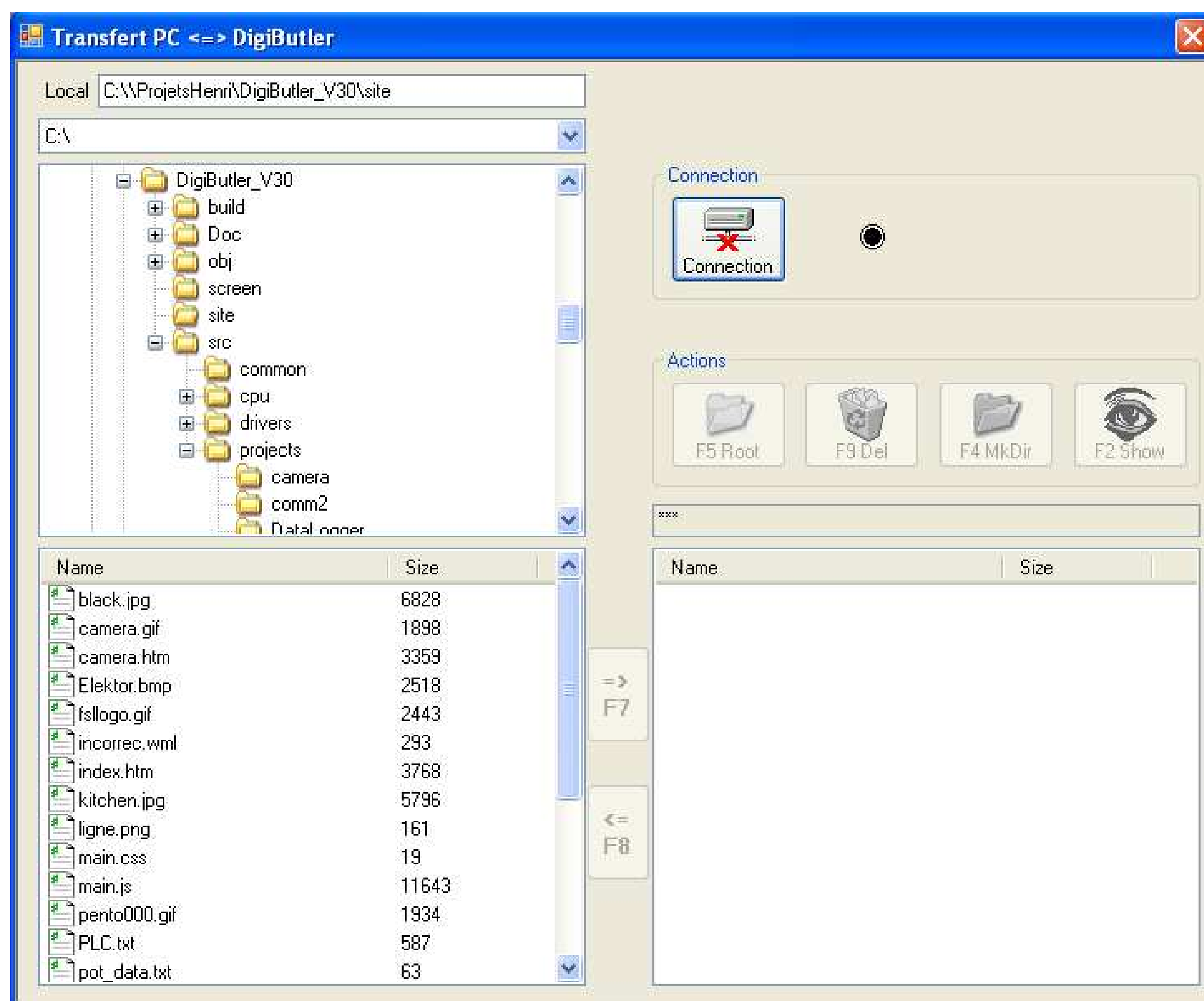


Figure 4

La moitié gauche est l'arborescence du PC.
Une modification du disque ou du répertoire en cours est sauvegardée avec les ***Settings*** pour afficher le même chemin au prochain lancement du programme.

Le message de modification apparaîtra donc.

Tous les boutons sont invalidés sauf celui de connexion avec DB.

Connecté :

Si les ***Settings*** Champs 1, 2 et 3 sont corrects, la connexion doit se faire correctement pour donner :

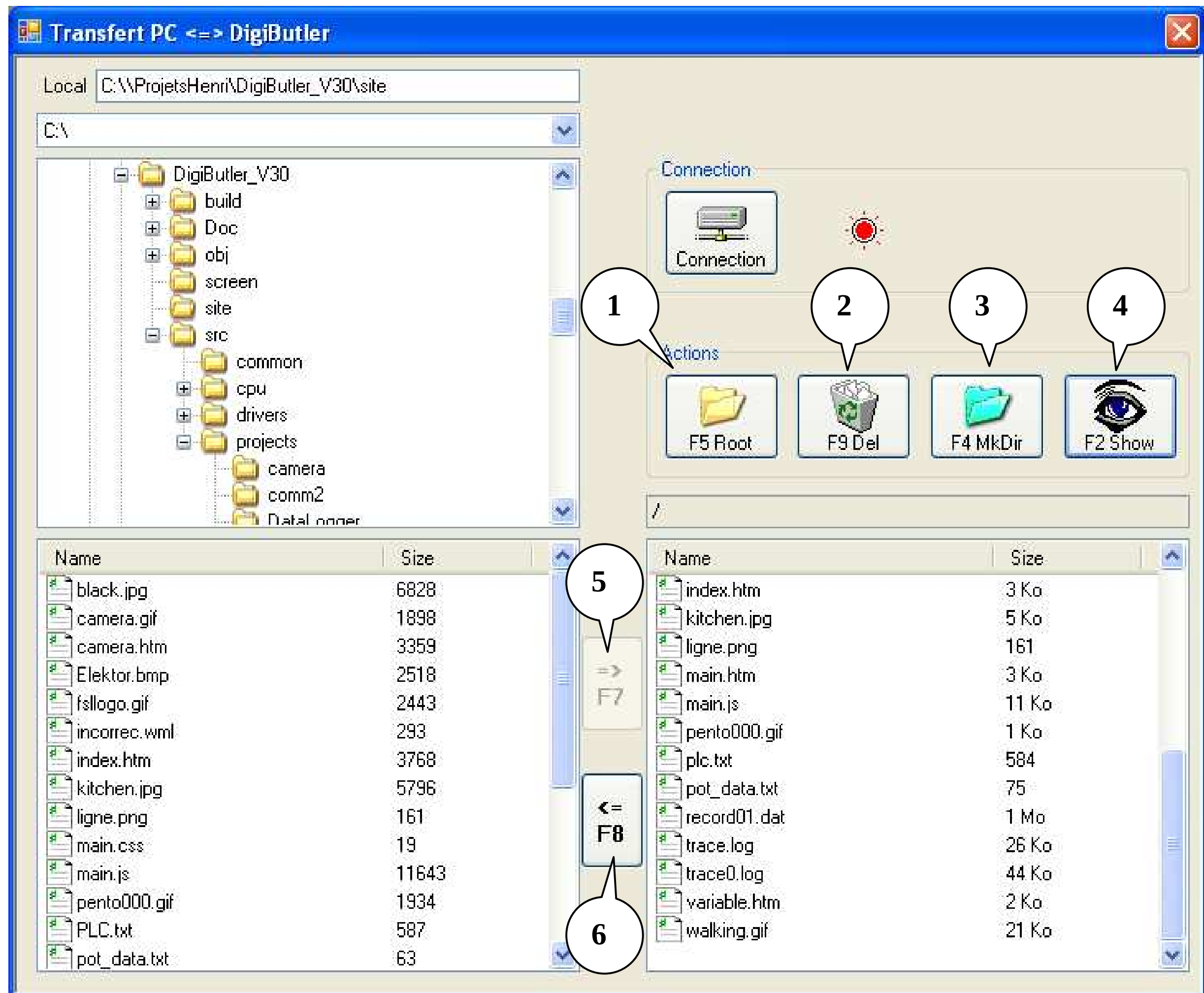


Figure 5

- Bouton 1 : Retourne à la racine (quand il y a des répertoires)
- Bouton 2 : Détruit le fichier touché (sur DB)
- Bouton 3 : Création d'un répertoire (sur DB)
- Bouton 4 : Charge le fichier touché avec l'exécutable (***Settings File Mappings***)
- Bouton 5 : Transfert le fichier sélectionné du PC vers DB
- Bouton 6 : Transfert le fichier sélectionné de DB vers PC

Le bouton 4 apparaît quand le fichier sélectionné a une extension connue.
Le bouton 5 apparaît quand le fichier sélectionné est coté PC.
Le bouton 6 apparaît quand le fichier sélectionné est coté DB.

DataLogger :

Fonctionnalité non implémentée sur ARM7

Utilisable uniquement avec Digibutler (ColdFire).

Déconnecté :

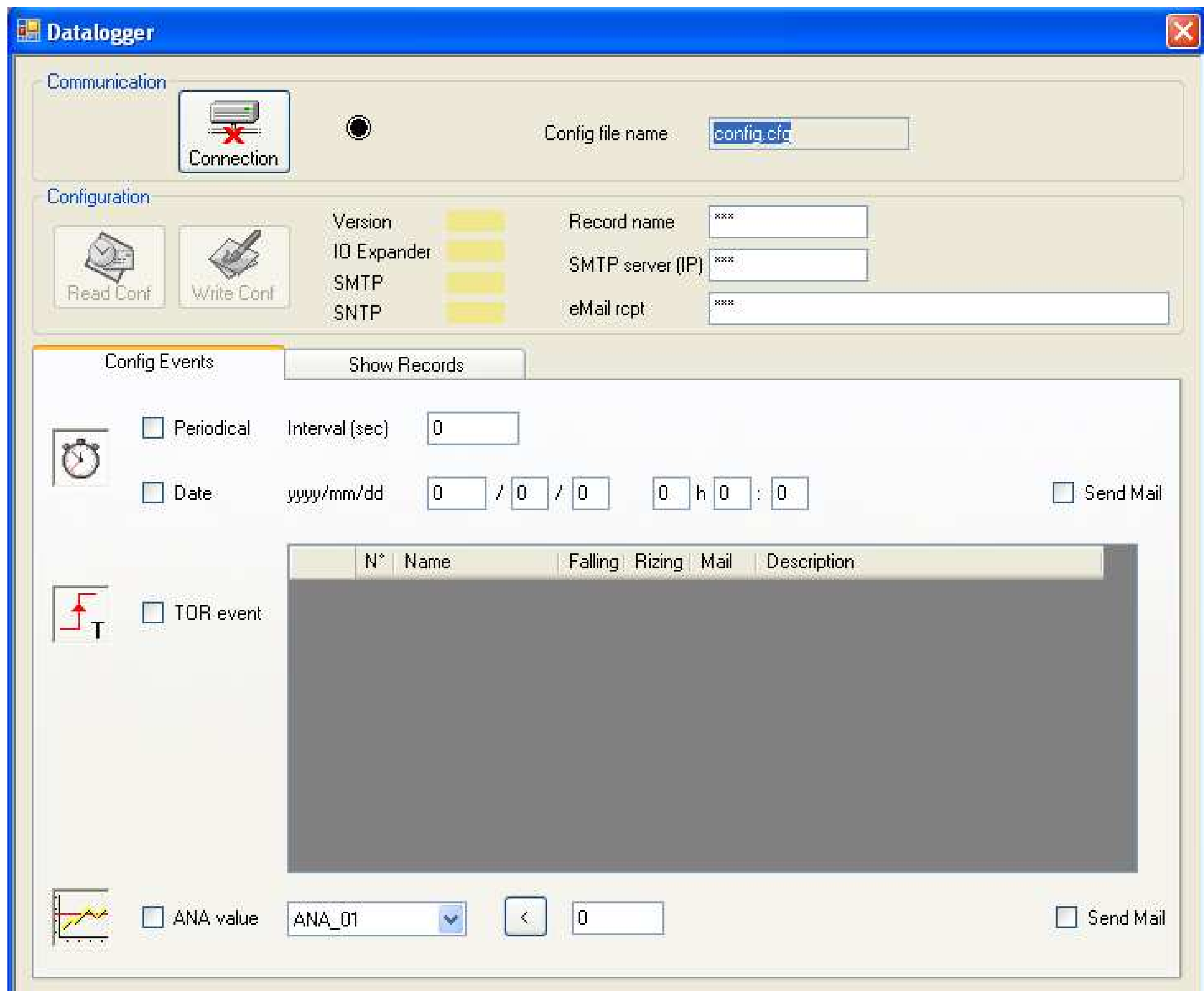


Figure 6

Connecté :

Une fois connecté, les boutons ***Read Conf*** et ***Write Conf*** deviennent valides.
Sont affichés la version du soft Datalogger, le nombre d'expanders et si les clients SMTP et SNTP sont activés sur DB.
Le client SMTP permet d'envoyer des eMails sur condition remplie.
Le client SNTP met l'horodateur à jour et permet donc d'utiliser un événement à date.

Figure 7

Lecture Configuration Datalogger :

Le bouton ***Read Conf*** va lire le fichier de configuration stocké sur DB.
Settings Datalogger Champ 11

Figure 8

Config générale :

- Champ 1 : Nom du fichier ou sont enregistrés les événements (fichier binaire).
- Champ 2 : Adresse de votre serveur SMTP (envoi de Mails).
- Champ 3 : Adresse du destinataire (envoi de Mails).

Config des événements :

- Champ 4 : Définit qu'il y a enregistrement périodique (champ interval).
- Champ 5 : Définit qu'il y a enregistrement à date (et envoi ou non d'un Mail).
- Champ 6 : Définit qu'il y a enregistrement sur variation d'une IO TOR
  Chaque IO est configurable pour générer un enregistrement sur front montant (rising edge), descendant (falling edge) ou les 2. De plus, on peut expédier un Mail.
- Champ 7 : Définit qu'il y a enregistrement quand une des 8 entrées analogique devient inférieure, supérieur , égale ou fifférente à une valeur. Avec envoi de Mail.

Ecriture configuration Datalogger :

Une fois modifié un des champs, on doit écrire sur DB avec le bouton ***Write Conf*** pour que la modification soit prise en compte.

Visualisation des enregistrements :

L'onglet ***Show records*** fait apparaître les éléments de consultation.

Figure 9

Une pression du bouton ***Numbers*** interroge DB pour connaître le nombre d'enregistrements présents dans le fichier (***Config générale champ 1***) en cours et permet ainsi la navigation.

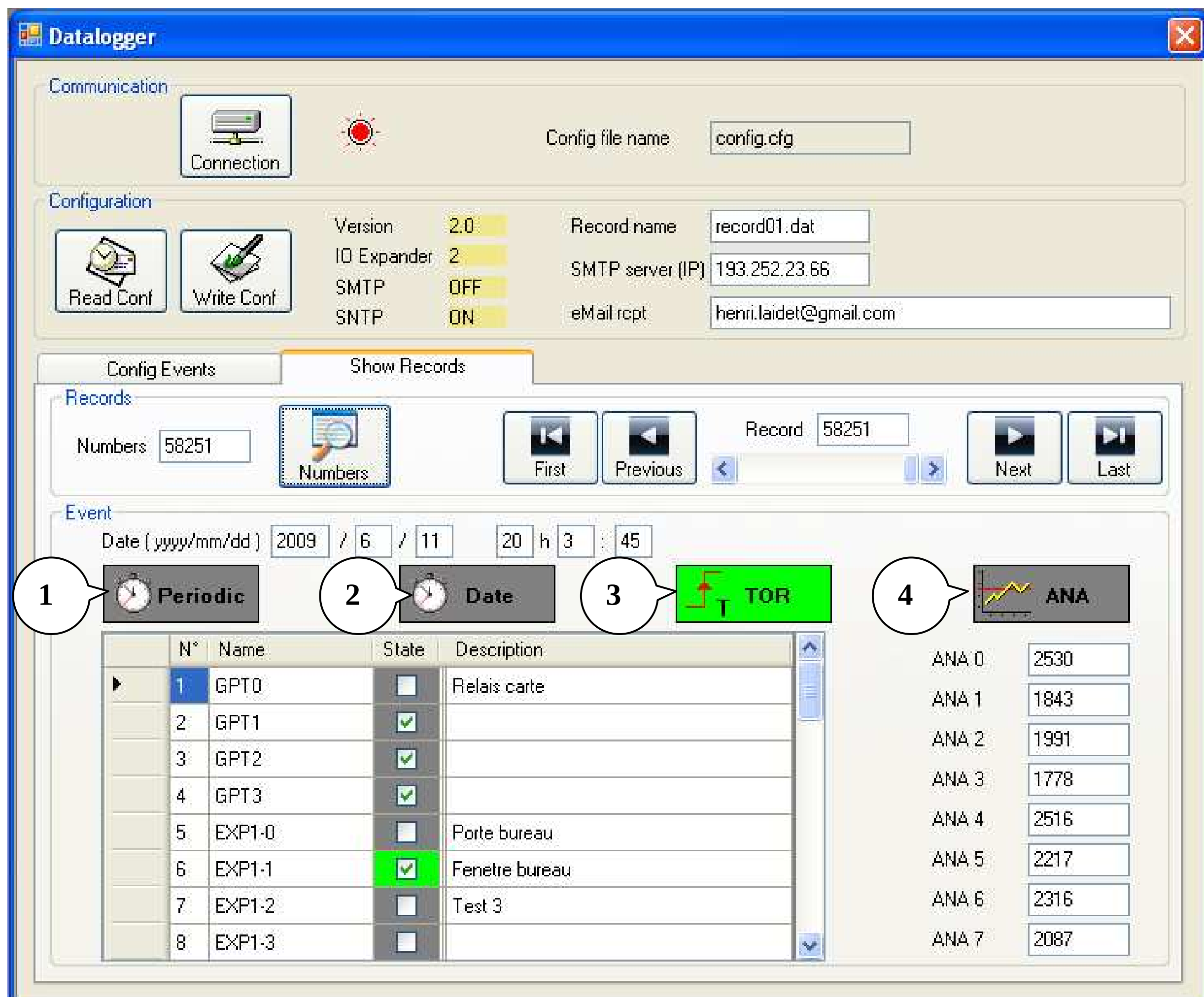


Figure 10

Les boutons ***First*** (premier enregistrement), ***Previous*** (enregistrement précédent), ***Next*** (enregistrement suivant) et ***Last*** (dernier enregistrement) nous déplacent dans le fichier. On peut aussi utiliser le slider ou saisir directement le numéro d'enregistrement visé.

Les enregistrements sont datés.

- Champ 1 : Vert indique un enregistrement périodique.
- Champ 2 : Vert indique un enregistrement à date.
- Champ 3 : Vert indique un enregistrement sur événement TOR.
- Champ 4 : Vert indique un enregistrement sur condition entrée ANA vraie.

Les événements peuvent être simultanés (ex : Périodique et TOR).
La période de travail de Datalogger sur DB est 1 seconde.

Ladder :

Fonctionnalité implémentée sur DigiButler et ARM7.

Sur DB (ou ARM7), un moteur PLC (Programmable Logic Controller) exécute du ***plccode*** généré ici.
Le moteur est activable avec ***#define HL_PLC 1*** (dans « ipport.h » sur DB ou « config.h » sur ARM7).
Il est appelé toutes les 5ms (chaque tick). La base de temps est donc de 5ms.
Le ***plccode*** est une liste d'objets (AND, OR, PUSH, POP) que le moteur interprète.

H_DigiPLC permet d'éditer un programme en mode graphique, de le compiler, de le transférer pour exécution sur DB (ou ARM7) et de le debugger (mode Monitoring).

Création :

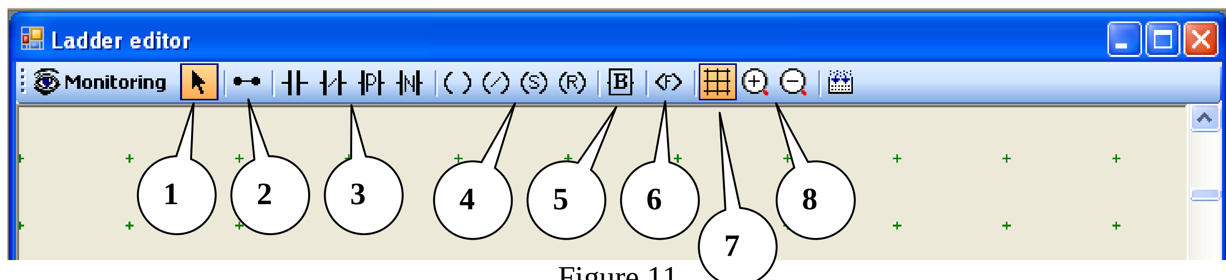


Figure 11

La création d'un programme se fait avec les outils champs 1 à 6

- Champ 1 : Flèche permet de sélectionner un objet sur la feuille et le déplacer
- Champ 2 : Ligne permet de relier les objets entre eux
- Champ 3 : Contacts d'entrée (Simple, Inversé, Front montant, Front descendant).
- Champ 4 : Sorties (Simple, inversée, Set, Reset)
- Champ 5 : Blocs (conditions spéciales)
- Champ 6 : Sorties spéciales (non booléennes)
- Champ 7 : Place ou non la grille sur la feuille.
- Champ 8 : Grossit ou réduit le graphique.

Les outils :

Flèche sélection et/ou déplacement des objets.

Tracé de segment.
La grille représente les points de connexion entre les cellules.
Avec cet outil, un petit carré se déplace sur la grille en suivant le curseur de la souris.
Un 1er click positionne l'origine du tracé (le carré reste sur place).
Le second click trace entre les 2 points en commençant par le segment horizontal

Contact simple : la liaison est établie si le bit est vrai.

Contact Not : la liaison est établie si le bit est faux.

Front montant : la liaison est établie si le bit passe de d'état faux à l'état vrai.

Front descendant : la liaison est établie si le bit passe de d'état vrai à l'état faux.

Sortie simple : le bit prend l'état du point d'entrée.

Sortie Not : le bit prend l'état inverse du point d'entrée.

Sortie Set : si le point d'entrée est vrai le bit est mis à vrai.
Sinon il reste inchangé.

Sortie Reset : si le point d'entrée est vrai le bit est mis à faux.
Sinon il reste inchangé.

Bloc : Fonction spéciales qui n'utilisent pas des booléens. Liste plus loin.

Sortie non bool : action sur des integers (MOVE, ADD, SUB, AND, OR…)

Extensions :

Un click droit fait apparaître quelques extensions :

Delete Object	F9
Delete All	F10
Insert Line	Insert
Delete Line	Delete
Refresh	F5
Check equ	F6
Properties	F4

Figure 12

- Delete Object : Efface l'objet séléctionné
- Delete All : Efface toute la feuille
- Insert Line : Insertion d'une ligne à la position courante.
- Delete Line : Efface la ligne à la position courante (si elle est vide)
- Refresh : Retrace la feuille
- Check equ : Contrôle si la partie du graphique est compilable
- Properties : Ouvre la fenêtre de propriétés de l'objet sélectionné

Les règles :

Le graphique est d'abord convertit en équations booléennes pour chaque sous ensemble.

Exemple : S1=(bit.bit2)+bit3

1. **Sorties :**
   Il ne doit y avoir qu'une seule sortie par équation (sinon impossible à formuler).
   Les éléments d'une équation associés à une sortie ne doivent pas être liés à une autre équation (encore impossible à formuler).
   La sortie doit être l'objet le plus à droite du graphique.
2. **Les lignes d'alimentation :**
   Elles ne sont pas matérialisées par un tracé vertical.
   Celle de gauche est le bord gauche de la feuille.
   Celle de droite est liée à l'objet le plus à droite d'une équation (donc de la sortie).
   Chaque équation peut finir à une position différente (elles sont traitées de façon isolées).

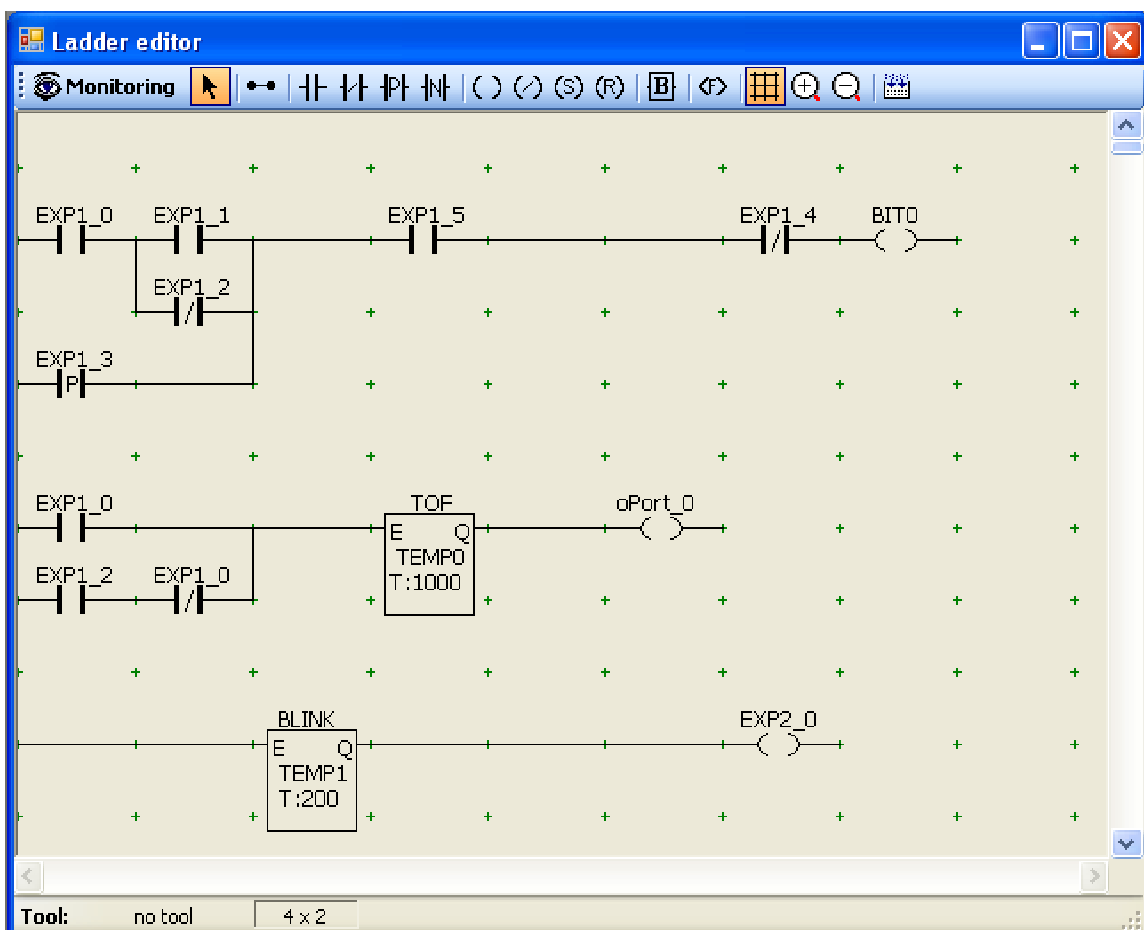


Figure 13

Mode opératoire :

On sélectionne un outil (type d'objet ex : Figure 11 champ 3, 4, 5 ou 6).
Le curseur de la souris prend la forme de l'objet sélectionné.
On se place dans la cellule visée et on clique.

Une fenêtre de propriétés est affichée

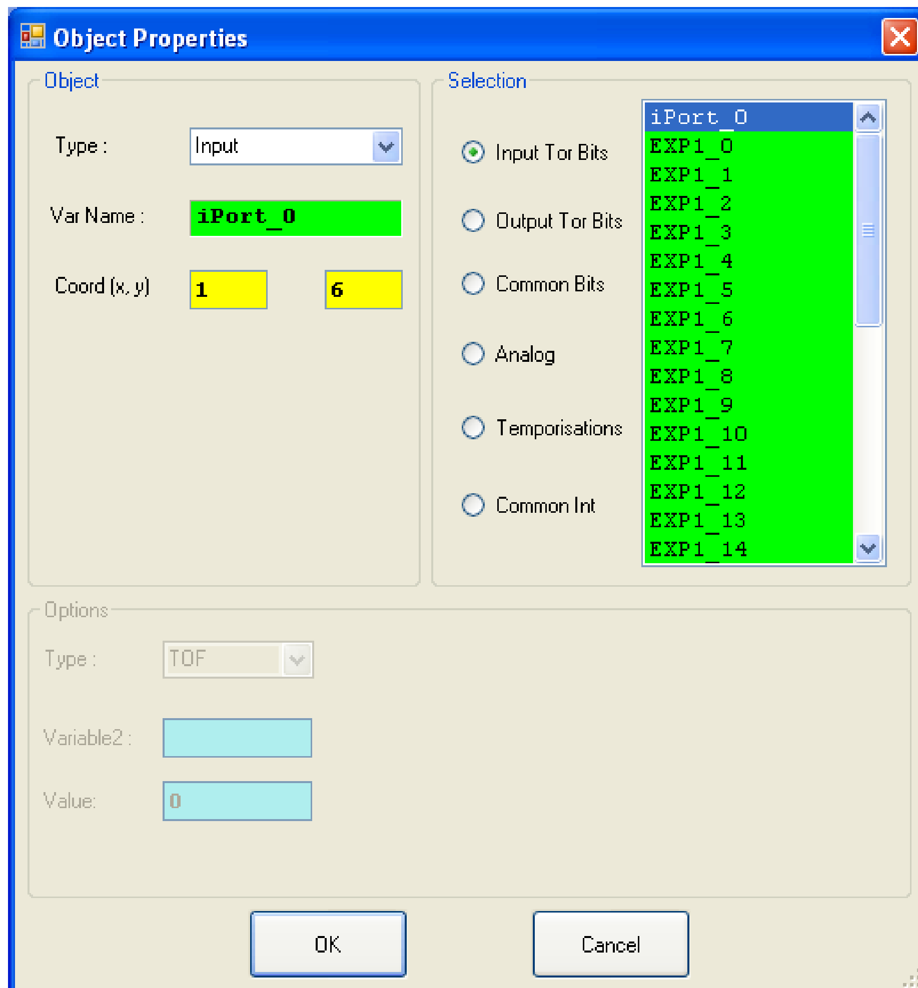


Figure 14

On peut choisir dans la liste de sélection :
Pour un contact (simple ou non) une entrée TOR une sortie TOR ou un Bit usage général.
Pour une sortie (simple ou non) une sortie TOR ou un Bit usage général.
Pour les Ports, on trouve iPort_x et oPort_x pour input et output car sur Digibutler les adresses de port (GPT0-3) ne sont pas les mêmes suivant la direction (MCF_GPIO_SETTA pour les bits en entrée et MCF_GPIO_PORTTA pour les bits en sortie).
Sur ARM7, on peut utiliser indifféremment iPort et/ou oPort
Le moteur PLC fait la différence.

Les blocs sont des conditions spéciales
Il y en a pour l'instant 8 sortes :

- Bloc TON : Tempo vraie si E valide depuis « temps »
- Bloc TOF : Tempo vraie si E valide ou E invalide depuis moins de « temps »
- Bloc TMP : Tempo vraie si E valide pas plus de « temps »
- Bloc BLINK : Clignotant

Les blocs de temporisation ont un petit graphique de chronologie.
La base de temps étant de 5ms, le timing est en 5ms.
On doit choisir dans la liste des temporisations.

- Bloc CMP_LT : Comparaison valeur1 < valeur2
- Bloc CMP_EQ Comparaison valeur1 = valeur2
- Bloc CMP_GT : Comparaison valeur1 > valeur2

Valeur1 peut être une entrée analogique, une temporisation ou un Integer usage général.

- Bloc HISTER : Devient vraie si valeur1 > (valeur2+hystérésis)
  Devient fausse si valeur1 < (valeur2-hystérésis)

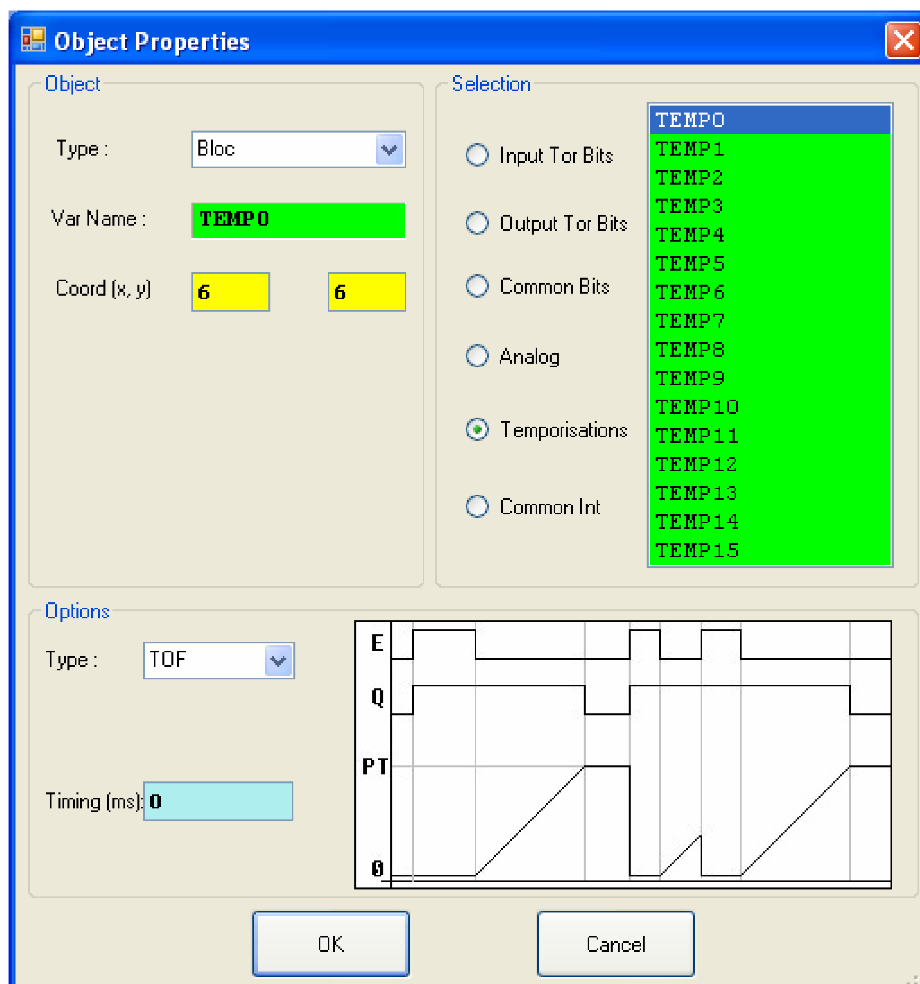


Figure 15

Après avoir renseigné la fenêtre de propriétés (bit, variable, timing…), l’objet est placé sur la feuille. On répète l’opération le nombre de fois nécessaires.
On utilise l’outil Ligne pour tracer les segments de liaison entre objets.
L’outil flèche permet :

1/ de déplacer les objets sur la feuille
2/ de modifier les propriétés d’un objet (changer bit, type, timing…)
Pour cela, on sélectionne l’objet (click sur la cellule) puis click droit (Figure 12) ou F4

Les fonctions sont des actions sur des entiers
Il y en a pour l'instant 8 sortes :

- Fonction MOVE : Variable = variable2
- Fonction ADD : Variable = variable + variable2
- Fonction SUB : Variable = variable - variable2
- Fonction AND : Variable = variable and variable2
- Fonction OR : Variable = variable or variable2
- Fonction XOR : Variable = variable xor variable2
- Fonction LSR : Variable = variable >> variable2
- Fonction LSL : Variable = variable << variable2

Les fonctions ont un petit graphique d'aide :

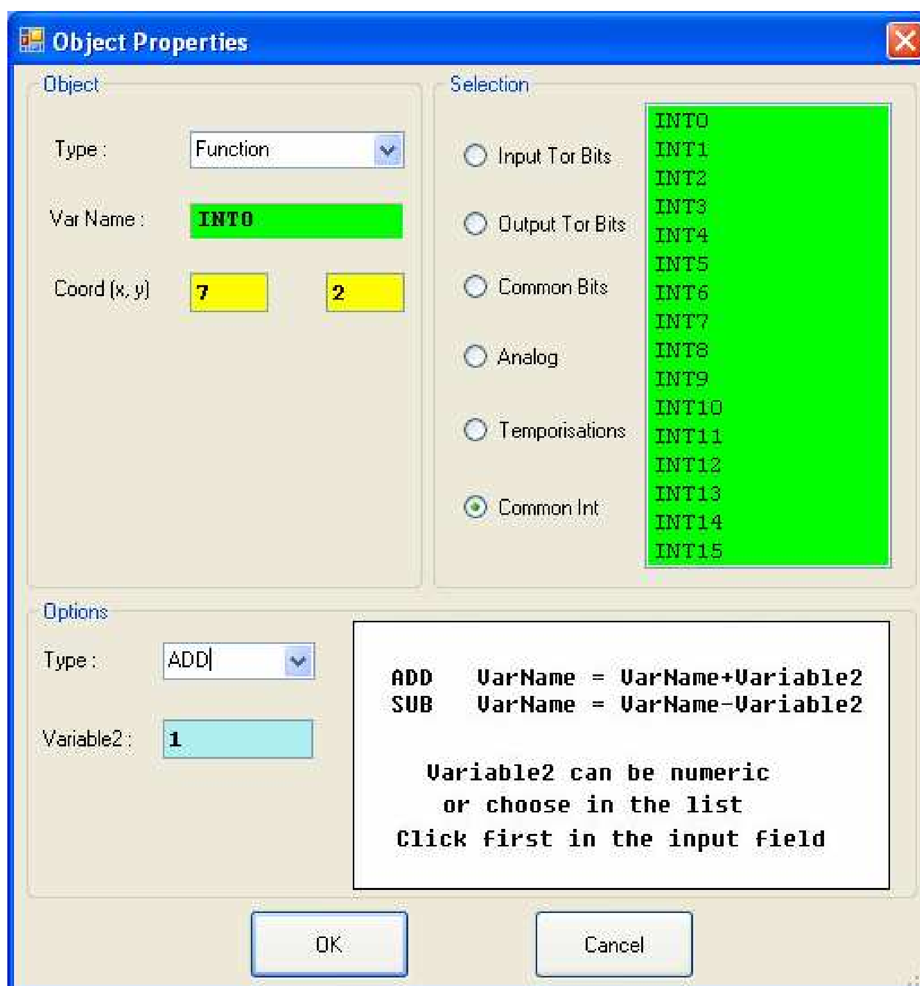


Figure 15-2

Pour utiliser les fonctions, il faut bien sûr que DB->PLC soit au moins en V1.1

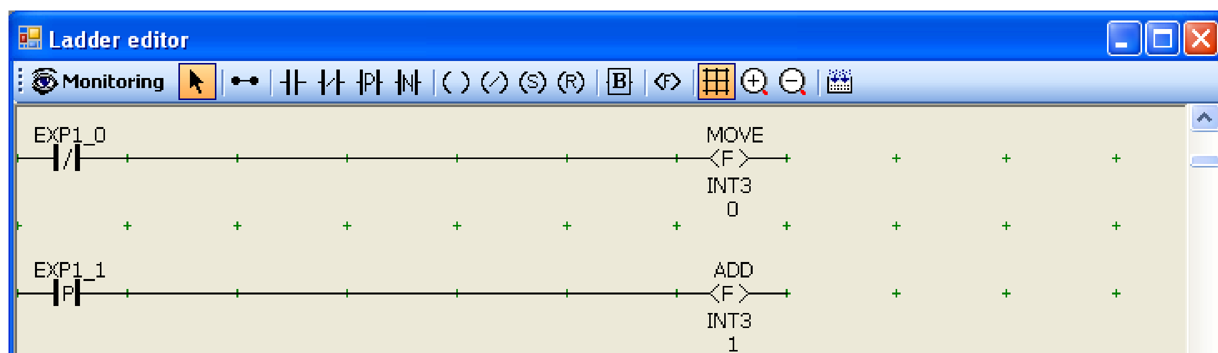


Figure 15-3

Compilation :

Une fois la création (ou modification) du graphique achevée, un click sur ***Build PLC file*** (Figure 16 : champ 1) lance la compilation qui ouvre une fenêtre de status.

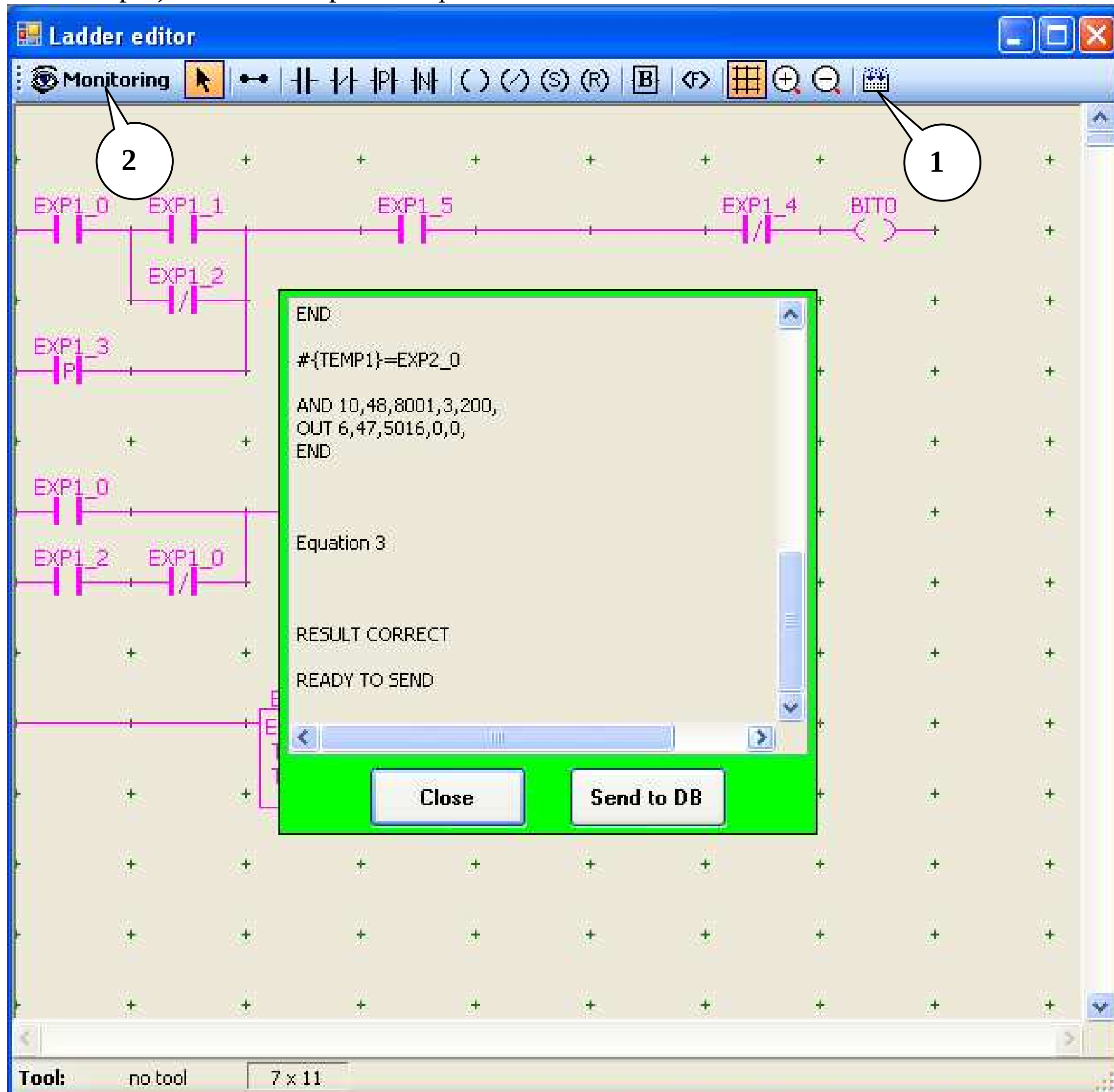


Figure 16

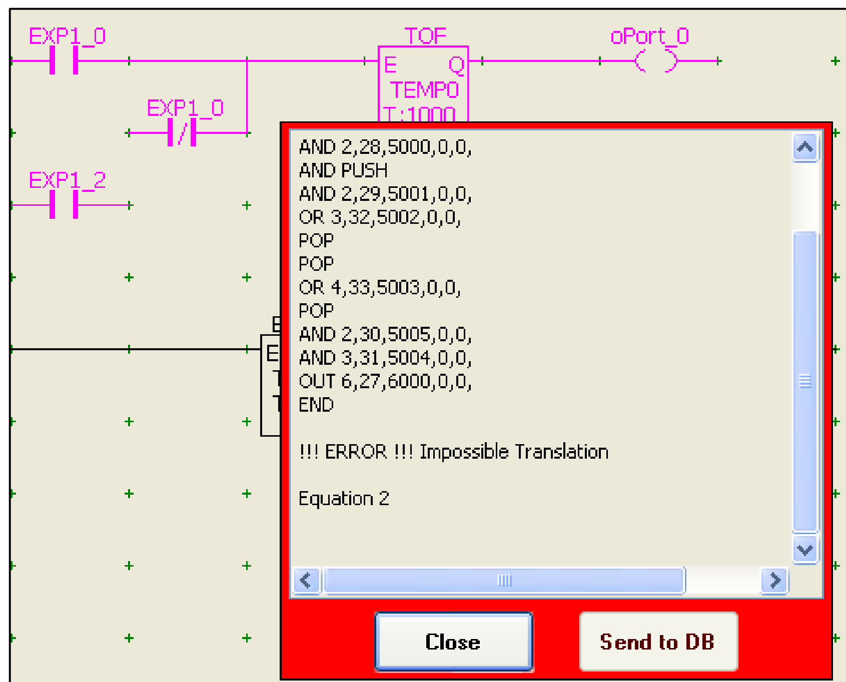
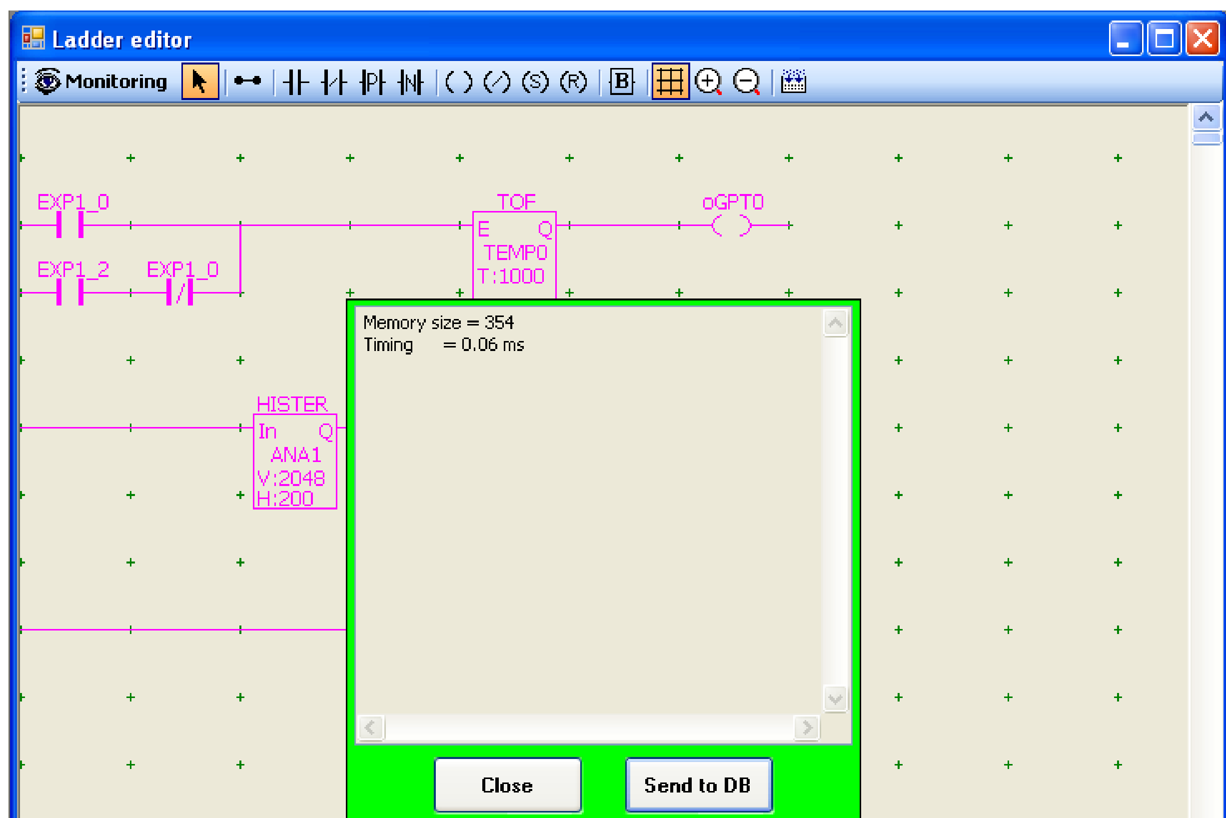


Figure 17

Si il y a une erreur, la fenêtre de status est rouge et le bouton ***Send to DB*** est invalidé (Figure 17). Sans erreur, la fenêtre de status est verte et le bouton ***Send to DB*** est valide (Figure 16).

Dans le cas où il n'y a pas d'erreur, un click sur ***Send to DB*** envoie le fichier compilé à la cible (DB ou ARM7).
La fonction PLC de la cible exécute ceci :
1/ Arrêt du programme Ladder en cours + libération de la mémoire utilisée.
2/ Lecture et décodage du fichier reçu (PLC.TXT) pour connaître la RAM nécessaire.
3/ Chargement du fichier (décodé) en mémoire.
4/ Exécution du programme une fois pour mesurer son temps d'exécution (en 10µs).
5/ Retourne taille et timing puis se met en standby.


Ladder editor
Monitoring
EXP1_0
EXP1_2
EXP1_0
TOF
E
Q
TEMP0
T:1000
oGPT0
HISTER
In
Q
ANA1
V:2048
H:200
Memory size = 354
Timing = 0.06 ms
Close
Send to DB
Figure 18

Exécution :

Un click sur ***Monitoring*** (Figure 16 champ 2) ouvre la section de contrôle à gauche de la feuille.

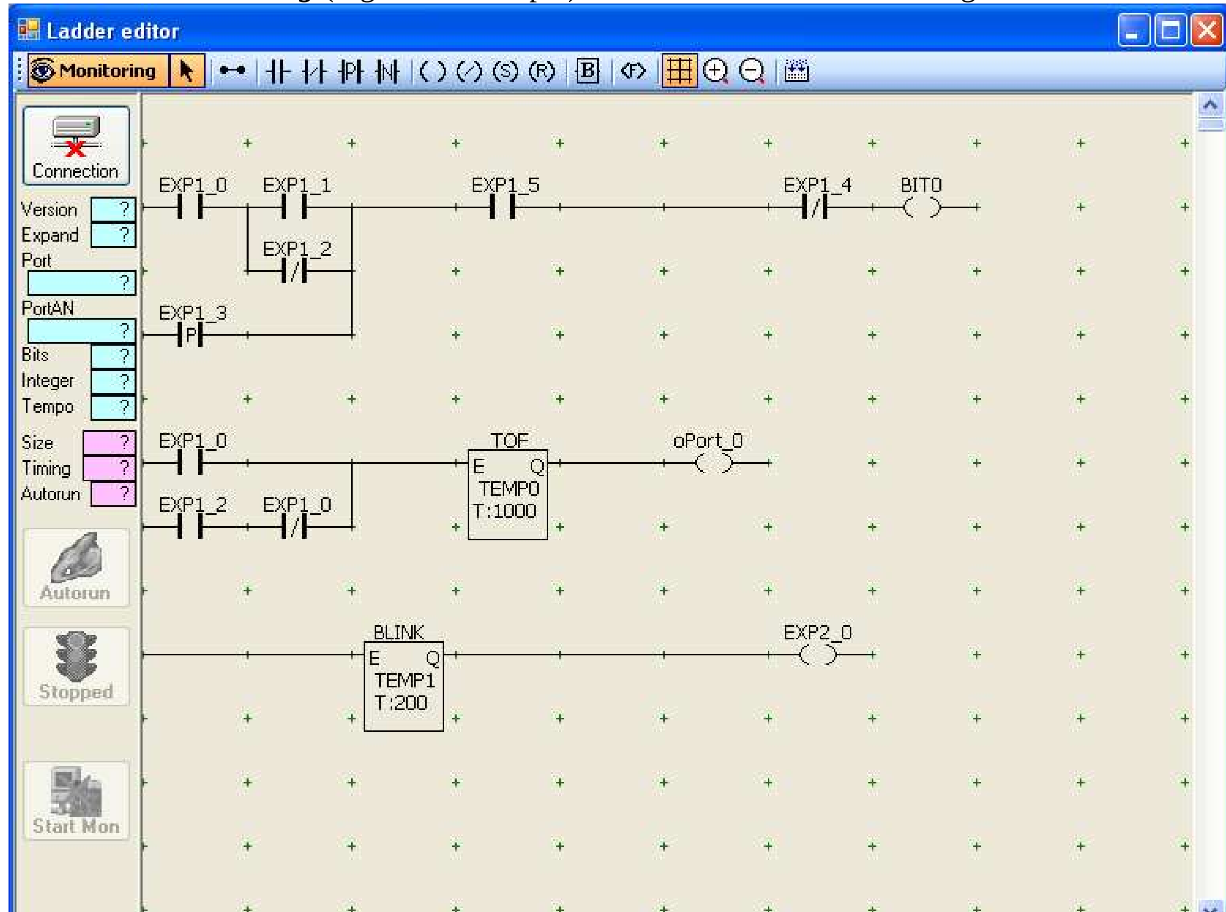


Figure 19

A la connexion PC⬄DB, la cible répond :

1/ Sa version
2/ Le nombre d'expanders qu'elle possède
3/ Le masque des bits du port principal (GPT0-3 sur DB ou port 0-31 sur ARM7)
4/ Le masque des bits du PortAN utilisés en IO TOR (0 sur ARM7)
5/ Le nombre de bits d'usage général
6/ Le nombre d'entiers d'usage général
7/ Le nombre de tempos

**** si les champs 2 à 7 ne correspondent pas aux ***Settings***, il y a demande de mise à jour ****

8/ La taille du programme PLC chargé
9/ Le temps d'exécution de celui-ci
10/ Si le programme se lance automatiquement au reset

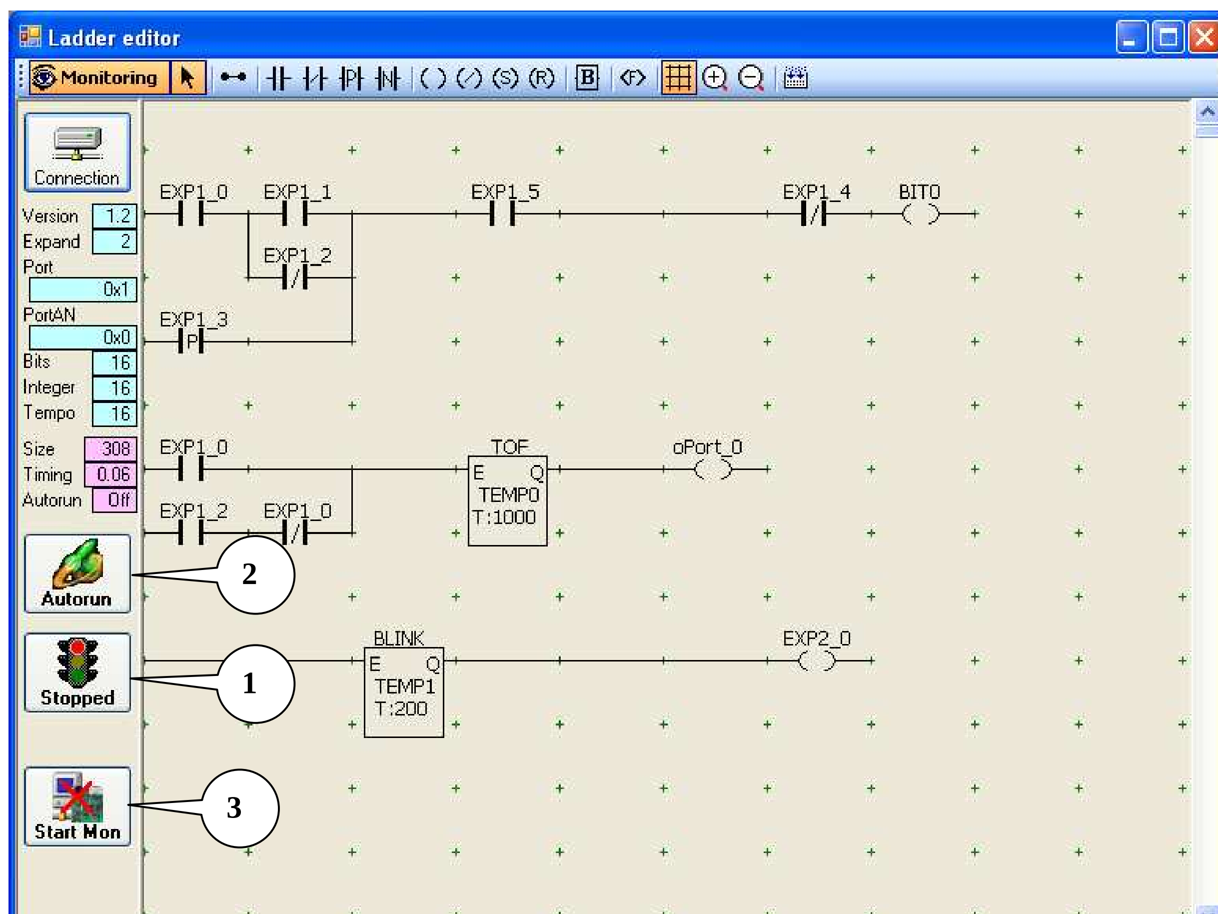


Figure 20

Le bouton Figure 20 champ 1 active ou arrête le moteur PLC pour debug.
Une fois le programme mis au point, le bouton ***Autorun*** donne la propriété de lancement automatique au programme PLC.
Cette propriété d'autorun est une ligne ajoutée au fichier « PLC.TXT » par DB lui-même..
Une action sur le bouton ***Autorun*** est donc nécessaire après un téléchargement (***Send to DB***) car la compilation ne place pas cette propriété active. Au cas où il y aurait une erreur bloquante.

Monitoring :

Une pression sur ***Start Mon*** (Figure 20 champ 3) lance la communication avec la cible (DB ou ARM7).

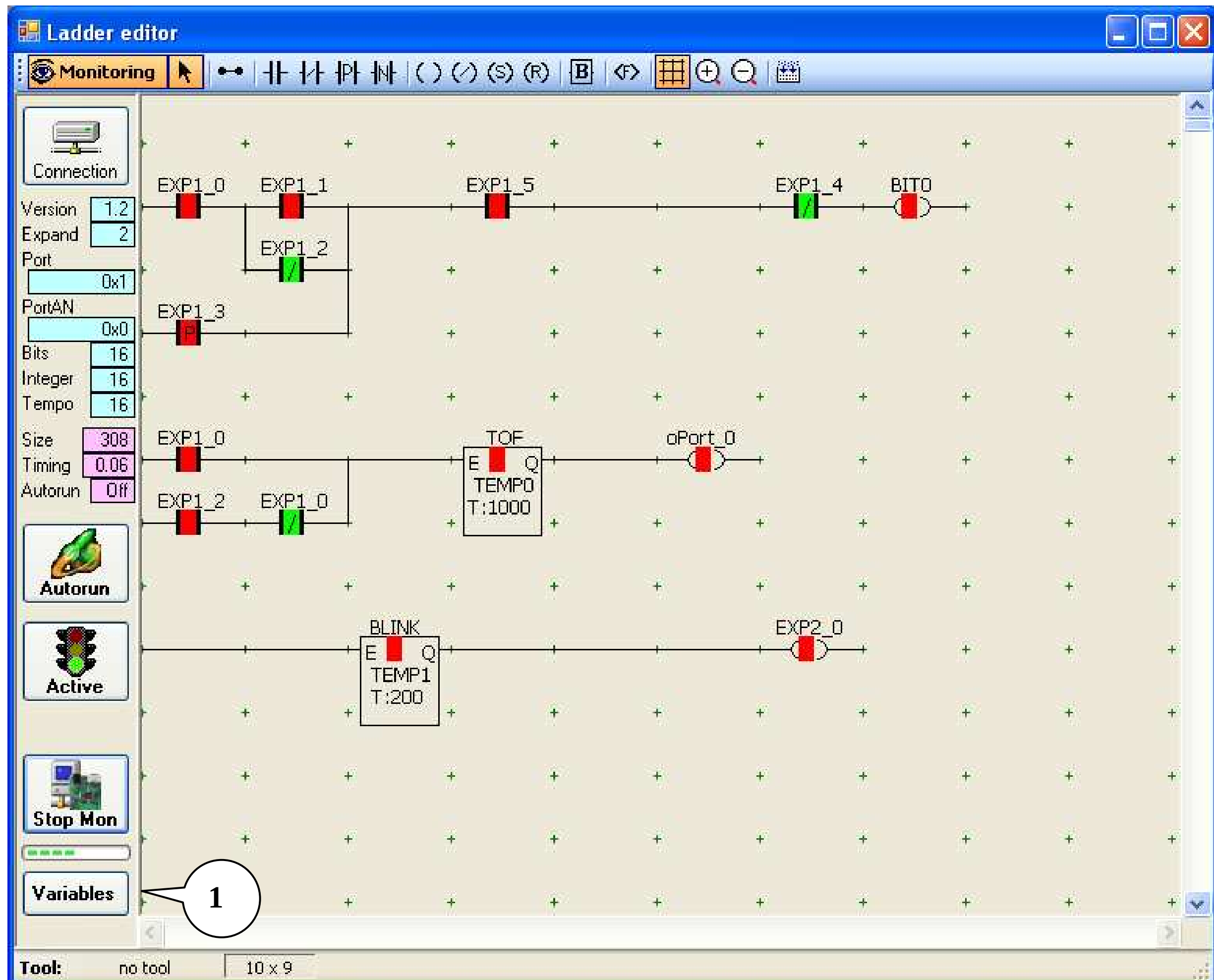


Figure 21

Quand on est en monitoring, on ne peut plus modifier le graphique mais seulement se déplacer.
Le programme interroge la cible sur l'état des objets en cours de traitement.
A la réponse de la cible, le graphique est animé en vert et rouge en fonction de l'état de l'objet.
Un bouton ***Variables*** apparaît (Figure 21 champ 1).
Une pression sur ***Variables*** fait apparaître une zone de suivi des IO à droite de la feuille.

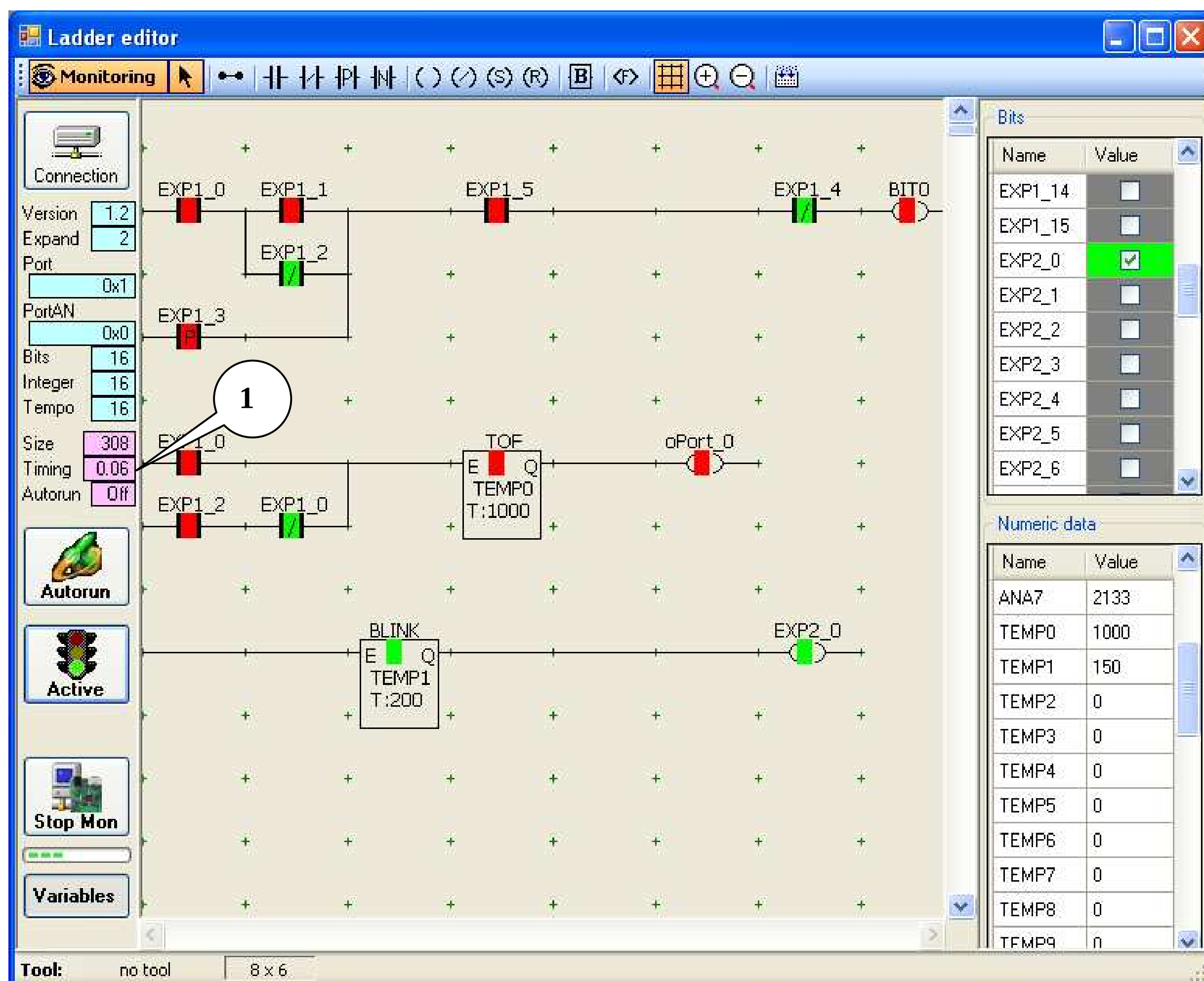


Figure 22

Performances :

On peut remarquer dans l’exemple Figure 22 champ 1 que DB exécute ce programme en 60 microsecondes.

Compatibilités Plates-formes Digibutler, ARMEE et Sceptre :

Différences

	Digibutler (K3)	ARMEE	SCEPTRE
Microprocesseur	ColdFire MCF52231 (59)	ARM7TDMI	ARM7TDMI
Quartz	25 MHz	14.745 MHz	12 MHz
Ports TOR	GPT0-3 et PortAN0-7	Port0-31	P0:0-31, P1:16-31
Ports ANA	PortAN0-7	-	AD0:0-7
Communication	Ethernet TCP/IP	Port série 115 kbauds	Port série (Bluetooth)
Carte SD	PortAN4-7	Port 18-21	SPI1
Expanders	Facultatif sur GPT0-3	Facultatif sur Port 3-7	Facultatif I2C ou SPI
Horloge RTC	Interne mise à l'heure SNTP	-	Interne
Compilateur	CodeWarrior 7.1	WinARM-20060606	
Programmation µP	BDM (TBLCF, P&E …)	Port série (Flash utility)	Flash Magic

Communication PC⇔Cible

Digibutler communique en Ethernet TCP/IP alors que ARMEE n'a que des lignes séries.
Sceptre a uniquement des lignes séries dont une sur Bluetooth (Profil liaison série).
La communication PC⇔DB via plusieurs ports TCP a été supprimée pour offrir à ARMEE les mêmes fonctionnalités.

Transfert fichiers sur carte SD

Tunnel FTP (client sur PC et serveur sur cible) assure l'interface.
Le Tunnel FTP existe sur les 3 plates-formes.
DB possède Tunnel FTP dans une requête http.
ARMEE possède un shell sur la ligne série pour les commandes moniteur dans lequel le Tunnel FTP est inséré (Header+size+data+CRC16).
Sceptre possède un shell sur la ligne série USB pour les commandes moniteur. Le Tunnel FTP est sur la ligne série Bluetooth.

Toutes les fonctionnalités de transfert PC⇔Carte SD sont présentes sur les 3 plates-formes.

Datalogger

Non implémenté sur ARMEE (pas de RTC) ni sur Sceptre.

PLC

Grâce à la suppression des connexions TCP multiples, la majorité des fonctions PLC existe sur les 3 plates-formes.
ARMEE n'a pas d'entrées analogiques mais elles apparaissent dans la liste de choix (extension future).
ARMEE ne mesure pas le timing du programme et répond 0 (extension future).
Toutes les autres fonctions sont opérationnelles avec ou sans carte expander.

Sceptre a un accelerometre 3 axes relié aux entrées ana 0-2.
Sceptre a un thermometre one wire. La temperature est stockée dans l'entrée ana 7 (non connectée).